Hacia la obtención de un modelo de base de datos relacional desde sistemas heredados: Un caso de estudio

Angélica Maldonado[1], Gilberto Gutiérrez[2], Angélica Caro[2], Alfonso Rodríguez[2]

[1] Cooperativa Eléctrica Los Ángeles Ltda.
Los Ángeles, Chile.
ange21@gmail.com

[2] Grupo de Investigación Pehuen
Departamento de Ciencias de la Computación y Tecnologías de la Información
Universidad del Bio-Bio
Chillán, Chile.
{ggutierr, mcaro, alfonso}@ubiobio.cl

Resumen. Hoy en día muchas organizaciones tienen Sistemas de Información Heredados que soportan una buena parte de la operación cotidiana de las mismas. En estos sistemas es posible encontrar uso de tecnología no relacional para administrar los datos, como por ejemplo archivos planos, bases de datos jerárquicas y de redes. Cuando se necesita evolucionar el software, esto puede significar grandes desventajas. Una manera de enfrentar esta situación es generar un nuevo esquema para los datos del sistema heredado que corresponda a un modelo relacional. En este artículo se presenta un caso de estudio en el cual se aplicó un método que permite crear esquemas de bases de datos relacionales a partir de diversas fuentes de información de un sistema de información heredado.

Palabras Claves: Sistemas Heredados, Datos Heredados, Modelo Relacional

1 Introducción

Un sistema de información heredado (LIS, Legacy Information Systems) puede ser crítico para la operación del negocio de una organización [4], algunos de ellos pueden operar las 24 horas del día [6]. Estos sistemas suelen estar construidos en lenguajes de segunda o tercera generación tales como Assembler, COBOL o FORTRAN y constantemente deben ser mantenidos para poder ajustarse a nuevas necesidades. Cuando se requiere una evolución más amplia del sistema generalmente se adopta una de las siguientes tres alternativas [2, 4]: (1) Migración (se mueve el LIS a un nuevo ambiente o plataforma más flexible, reteniendo la funcionalidad y los datos del sistema original), (2) Redesarrollo (que abandona el sistema y se sustituye por uno nuevo) y (3) Wrapping (que provee una nueva interfaz para el LIS o algún componente de éste, lo que permite mayor

accesibilidad desde otras aplicaciones). Sin embargo, independientemente de la solución adoptada, los datos deben ser conservados puesto que constituyen un importante activo para la organización.

Por otro lado, aún es frecuente en los LIS el uso de tecnología no relacional para administrar los datos, como por ejemplo archivos planos, bases de datos jerárquicas y de redes [5]. Esto presenta grandes desventajas cuando se requiere una evolución amplia de ellos. El modelo de bases de datos relacional ha sido y sigue siendo el modelo por excelencia para implementar sistemas de información operacionales y analíticos, e incluso también es usado con las nuevas tecnologías de las aplicaciones Web [5]. Las ventajas de la tecnología relacional son, entre otras, la solidez y simplicidad del modelo, su amplia difusión y el hecho de contar con un estándar que garantiza su continuidad en el mercado a través de variados productos como por ejemplo: Oracle, Postgres y SQL Server [9].

En este sentido, este artículo presenta un método que permite la generación de un modelo de base de datos relacional a partir de diversas fuentes de información de un LIS (descripción de archivos, código, interfaces, etc.). Esto, como primer paso de una futura migración de datos del LIS. Para explicar e ilustrar el uso del método, se aborda el caso de estudio de un LIS desarrollado en COBOL.

El resto del artículo se encuentra organizado de la siguiente forma. En la Sección 2 discutimos los trabajos relacionados. El método propuesto para la obtención del modelo relacional es presentado en la Sección 3. La Sección 4 presenta el caso de estudio y los resultados obtenidos. Finalmente, en la Sección 5 se presentan las conclusiones y trabajos futuros.

2 Trabajos Relacionados

En la literatura diversos trabajos abordan la necesidad de migrar los datos de un LIS hacia una base de datos relacional [5, 10, 12], siendo una de las primeras tareas la definición de las nuevas estructuras de datos. Esto porque uno de los mayores desafíos consiste, en proveer nuevas estructuras de datos que contengan toda la semántica contenida en los datos heredados y que además éstas no hereden características propias de la tecnología del LIS [10], las estrategias que persiguen estos objetivos suelen clasificarse como “basadas en la semántica”.

En este sentido, a la hora de generar las nuevas estructuras de datos, el modelo de base de datos relacional es el preferido [1, 3, 5, 8, 10]. Efectivamente, este modelo ha sido por décadas el estándar de facto para construir sistemas de información operacionales y analíticos, e incluso sigue siendo efectivo con las nuevas tecnologías [5].

Respecto de cómo extraer el nuevo modelo de datos, algunos trabajos proponen el uso de ingeniería reversa. Es el caso de [1] y [11], donde el primero propone un método para extraer un modelo entidad relación desde una base de datos relacional. La información del modelo se obtiene por medio de las consultas SQL establecidas en la aplicación y en las instancias de las relaciones. El segundo, recupera el esquema de la base de datos heredada CODASYL (modelo de red) a partir de instrucciones DDL (Data Description Language).

El proceso es semi-automático y no considera normalización adicional del esquema de la base de datos relacional resultante. Por otro lado, Boronat et al. [3] proponen una estrategia en la cual un diseñador construye un esquema conceptual O-O semánticamente equivalente al LIS y luego a partir de éste se genera de manera automática el esquema relacional para la nueva base de datos.

En general, los enfoques revisados se centran en la extracción de la semántica y el nuevo modelo casi exclusivamente desde el modelo heredado. Aunque otros incorporan nuevas fuentes de información como lo son el código, interfaces y documentos; tal es el caso de [10].

Para nuestro trabajo, que aborda la generación de las nuevas estructuras de datos, hemos adoptado una estrategia basada en la semántica, considerando las distintas fuentes de información del LIS e incluyendo a los trabajadores asociados a él (usuarios, mantenedores, expertos, etc.) en gran parte del proceso de generación del nuevo esquema. Adicionalmente, abordamos la normalización del nuevo esquema generado.

3 El método M-FF2RDB

La necesidad de generar una nueva estructura de datos antes de la migración de un LIS, y considerando una perspectiva en que la solución sea planteada como un método repetible y evaluable, nos ha movido a formular M-FF2RDB (del inglés Method-Flat_File to Relational_Data_Base). Este método permite, por medio de un conjunto de etapas, trabajadores, herramientas y artefactos, crear modelos de datos normalizados tomando como punto de partida los meta-datos o meta-atributos de un conjunto de archivos planos que forman parte de un LIS.

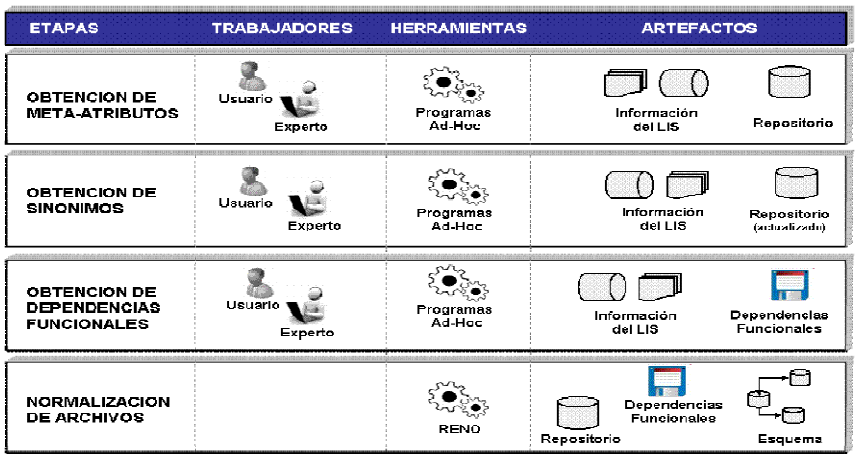

Fig. 1: Vista general del método M-FF2RDB

El método M-FF2RDB, que se presenta gráficamente en la Fig. 1, está compuesto por cuatro etapas que en conjunto permitirán, partiendo de un conjunto de archivos planos y la información en relación a ellos, obtener un modelo de bases de datos relacional. Las etapas del método son:

- OBTENCIÓN DE META-ATRIBUTOS: En esta etapa se lleva a cabo una recopilación de la información disponible en el LIS en relación con los archivos y sus descripciones. Los trabajadores, experto(s) y usuario(s), establecen el origen de esta información, ya sea que esté en documentación no disponible en medios magnéticos y/o en programas fuentes o catálogos con la descripción de archivos. Las estrategias de obtención de los meta-atributos contemplan el trabajo manual de los trabajadores y el uso de herramientas como programas ad-hoc que permitirán buscar en medios magnéticos la información y permitirán realizar la actualización de un Repositorio. El artefacto asociado a esta etapa son el Repositorio, que contendrá la información recuperada y servirá de base para las etapas siguientes, y la Información del LIS disponible en medios magnéticos y no magnéticos. El objetivo principal de esta etapa es la creación del Repositorio, descrito en la Fig. 2 y en donde se puede distinguir las entidades principales: Sistemas, Programas, Archivos y Atributos.

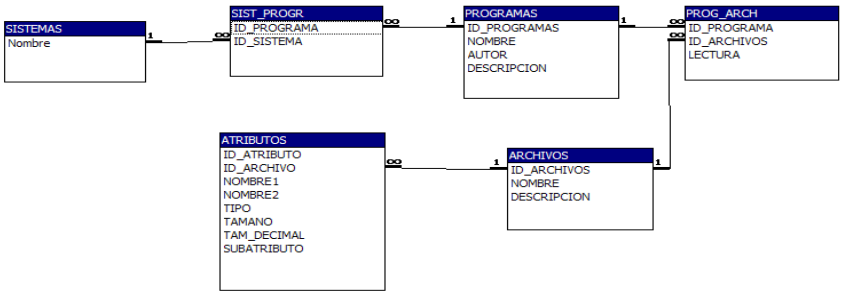

Fig. 2: Repositorio de meta-atributos

- OBTENCIÓN DE SINÓNIMOS: Esta etapa corresponde a la depuración de los atributos identificando y eliminando los duplicados que puedan existir. El criterio empleado para la identificación de sinónimos fue establecer porcentajes de coincidencia entre atributos basados en el nombre, tipo y tamaño del atributo. De esta forma, se tiene que aquellos atributos que tengan igual nombre, tipo y tamaño serán candidatos a sinónimos en un 100%. Los que tengan igual nombre y tipo tendrán un 75%, los que tengan igual tipo y tamaño tendrán un 50% y finalmente, los que sólo tengan igual nombre tendrán un 25%. La determinación de candidatos a sinónimos se hace por medio de programas ad-hoc que verifican las condiciones de coincidencia. Los trabajadores en esta etapa son usuario(s) y experto(s), quienes tendrán que decidir acerca del conjunto final de sinónimos. Los artefactos son el Repositorio (actualizado) y la Información del LIS (en formato magnético y no magnético).

- OBTENCIÓN DE DEPENDENCIAS FUNCIONALES: Esta etapa consiste en crear un almacén de datos que contenga las dependencias funcionales. La obtención de éstas se hace en forma automatizada (con programas ad-hoc que generan las dependencias funcionales deducibles a partir de los atributos claves/índices de los archivos) y en forma manual. En la parte automatizada se recurre la descripción de los archivos, sus claves y modos de acceso asociado. Los trabajadores de esta etapa son usuario(s) y experto(s). Los artefactos son la Información del LIS y un conjunto de dependencias funcionales.
- NORMALIZACIÓN DE ARCHIVOS: Esta etapa consiste en obtener un esquema de base de datos relacional normalizado. Esta etapa no requiere trabajadores pues está totalmente automatizada. Para ello, y a modo de experimentación, se emplea una herramienta llamada RENO [7] que permite, mediante la aplicación de los algoritmos de manejo de dependencias funcionales y de normalización la obtención de esquemas relacionales. Los artefactos son el Repositorio, el conjunto de dependencias funcionales y el Esquema relacional.

Una vez que hemos definido este método lo aplicamos a un caso de estudio real, con el objetivo de hacer una validación y mejora del mismo. El caso de estudio se llevó a cabo en una empresa de distribución de energía eléctrica rural, durante los últimos meses del año 2008. En la sección 4 se realizará una descripción más detallada de éste.

4 Caso de Estudio

Para la descripción del caso de estudio, esta sección se ha divido en cuatro partes: la caracterización del LIS, la descripción de los problemas que actualmente existen en relación con él, la aplicación del método y las lecciones aprendidas.

4.1 Caracterización de la aplicación heredada

El caso de estudio corresponde a una aplicación computacional que soporta el negocio de la Cooperativa de Electricidad de Los Ángeles Ltda., (Coopelan). Los sistemas que la conforman permiten administrar la distribución de energía eléctrica a sus socios y paulatinamente se han ido expandiendo para controlar la comercialización de otro tipo de bienes y servicios para sus asociados.

El conjunto de LIS que dan soporte a la cooperativa, que desde ahora denominaremos ACC (Aplicación Computacional de Coopelan) se encuentra desarrollado en COBOL. Estos sistemas datan del año 1985 en que fueron desarrollados para otra empresa del mismo rubro. Por la similitud de los procesos de negocio de ambas cooperativas estos sistemas fueron adquiridos por Coopelan en el año 2001.

En cifras, la ACC está conformada por 1.696 programas, agrupados en 25 sistemas, que interactúan a su vez con 387 archivos, que en total tienen 5.263 atributos. El detalle de los sistemas que actualmente se encuentran en operación se presenta en la Tabla 1, ésta

describe el sistema e indica el número de programas y archivos que utiliza.

Tabla 1. Sistemas de ACC.

Sistema	Descripción	Arch	Prog
S-1	Control de auditoría sobre los archivos más importantes	2	2
S-2	Control de pagos con actualización automática de los sistemas relacionados	55	60
S-3	Genera compensaciones por interrupciones de servicio eléctrico	4	5
S-4	Registra y mantiene la información de compras a proveedores	24	28
S-5	Registra los movimientos contables exigidos por la ley tributaria.	24	106
S-6	Control de la cartera de clientes que tiene crédito en la empresa	65	132
S-7	Registro las ventas crédito en las cuentas corrientes de clientes	42	70
S-8	Registro de empalmes eléctricos efectuados por la empresa.	3	7
S-9	Manejo de la información de los consumos de energía y su cobro	67	536
S-10	Facturación especial (empresas, energía activa y reactiva)	17	31
S-11	Facturación de ventas en locales comerciales de la empresa	56	93
S-12	Control de transporte de mercaderías (ventas a clientes)	7	6
S-13	Control boletas de honorarios	8	13
S-14	Administración de consultas de clientes (directa o por aproximación)	16	23
S-15	Inventario de las mercaderías (materiales eléctricos para proyectos eléctricos)	45	189
S-16	Control de la construcción de líneas de media tensión	32	67
S-17	Registro las pruebas y datos de los medidores de clientes	9	12
S-18	Registro de notas de crédito de la empresa	20	13
S-19	Registro y mantención de ordenes de compra hacia proveedores	13	35
S-20	Manejo de precios y descuentos especiales de artículos en oferta	5	5
S-21	Control de correspondencia interna	10	14
S-22	Manejo de proyectos eléctricos de nuevos clientes	63	180
S-23	Refacturación de consumos eléctricos a clientes por cobros indebidos o repactados.	13	8
S-24	Manejo de todos los valores de la empresa	32	53
S-25	Control de libros contables para presentación tributaria	11	8

Debido al proceso de adaptación propio que sufre el software que ha sido adquirido a otra empresa, la ACC no se usa en su totalidad. Actualmente, sólo se utilizan 218 de los 387 archivos, es decir hay un total de 169 que no se utilizan en ningún momento pero que aún están presentes.

En relación con la documentación existente se cuenta con los programas fuentes en COBOL y un catálogo que contiene información descriptiva de los archivos y de las rutas de acceso de éstos. Básicamente el catálogo contiene dos tipos de archivos que sirven de soporte para el manejo de los archivos de la ACC; los archivos ".FD" que contienen la estructura de los atributos de cada archivo, describen las claves primarias, tamaño de los atributos y sus tipos y los archivos ".SL" que contienen la ruta de los archivos ".FD", las claves y modos de acceso asociado (secuencial, indexado, etc.).

4.2 Problemática actual con la ACC

Durante años Coopelan ha trabajado con el sistema de archivos provisto por la ACC, sin embargo, en la actualidad se enfrenta a la necesidad de independizar los programas de sus

datos, para de este modo mejorar el manejo de los mismos, mejorar la consistencia en la información y lograr, además, independencia entre la lógica de negocio y la persistencia.

Los problemas más frecuentes que se tienen hoy día en relación con los archivos son: (i) la redundancia e inconsistencia de los datos ya que éstos se encuentran en diferentes formatos y duplicados en diversos archivos, (ii) dificultad de acceso a los datos debido a que existen aplicaciones que controlan de manera independiente el acceso a los datos, (iii) dificultad para utilizar los archivos de datos desde nuevas aplicaciones o nuevos programas, (iv) la actualización simultánea no se controla adecuadamente y suele producir información inconsistente, (v) resulta complicado restringir el acceso a los archivos y sus datos y, finalmente, (vi) debido a la no existencia de procedimientos claros que controlen el ingreso de datos incorrectos o incompletos se producen problemas de inconsistencia.

A raíz de lo anterior, el mayor problema surge con la independencia entre los datos y la lógica de negocio. Dada la estrecha relación que existe, resulta difícil modificar los programas sin que los datos se vean afectados y viceversa. También debido a la antigüedad de los sistemas y a que no han sido creados de manera ad-hoc para esta empresa, algunos archivos tienen información que hoy ya no se maneja, o bien hay programas que no se utilizan pero que están presentes. Depurar la ACC de estos archivos y programas de forma manual implica un alto costo en recursos y, en cualquier caso, ello no garantiza la solución de todos los problemas mencionados.

Ante esta situación, se ha determinado que la solución más adecuada, como una forma de comenzar un proceso de modernización, es la transformación del Sistema de Archivos de la ACC en una Base de Datos Relacional.

4.3 Aplicación de M-FF2RDB

La presentación de aplicación de M-FF2RDB en Coopelan se hará por cada una de las etapas, explicando la forma en que fue desarrollado y los resultados obtenidos.

- OBTENCIÓN DE META-ATRIBUTOS: en esta etapa se realizó un catastro de toda la información disponible sobre los datos del LIS. La obtención de los meta-atributos se hizo en forma automatizada con programas diseñados ad-hoc. Para el diseño de estos programas se elaboraron pruebas previas orientadas a determinar la forma en que la información podía ser rescatada desde los programas fuentes y las descripciones ".FD" y ".SL". Los programas ad-hoc (escritos en Java) permiten obtener los sistemas, programas, archivos de la ACC y los atributos y los sub-atributos de cada uno de los archivos del sistema, toda esta información se almacenó en el Repositorio.
- OBTENCIÓN DE SINÓNIMOS: A partir de la información almacenada en el Repositorio se realizó el análisis de las coincidencias entre los atributos, tomando como referencia el nombre, tipo y tamaño de éstos. Como resultado sólo se encontraron 14 posibles sinónimos con un 50% de coincidencias. Estos fueron contrastados con el experto, quien confirmó como sinónimos los mismos 14 atributos.

- OBTENCIÓN DE DEPENDENCIAS FUNCIONALES: Las dependencias funcionales se obtuvieron mediante dos pasos. En el primero de ellos, se obtuvo un conjunto inicial de dependencias funcionales, utilizando la información del catálogo (archivos .FD y .SL) mediante la combinación de las claves primarias de los archivos con el resto de los atributos. En el segundo paso, el conjunto inicial de dependencias fue revisado en forma manual por los expertos, quienes lo verificaron y complementaron con nuevas dependencias. Por último, se preparó un archivo de texto con las dependencias en el formato que se muestra en la Tabla 2.

Tabla 2. Ejemplo de dependencias funcionales

RUT→MATERNO, RUT→NOMBRE1, RUT→NOMBRE2, RUT→PATERNO
MOVCTA-CTACORTA, MOVCTA-CODCTA, MOVCTA-CCOSTO, MOVCTA-LOCAL→MOVCTA-NUMCLIE
MAE-MEDIDOR→MAE-COESTAD, MAE-MEDIDOR→MAE-MARCAME, MAE-MEDIDOR→MAE-NRUEDAS, MAE-MEDIDOR→MAE-NUFASES, MAE-MEDIDOR→MAE-POTEINS, MAE-MEDIDOR→MAE-PROP-MEDIDOR

- NORMALIZACIÓN DE ARCHIVOS: Para la normalización de los archivos se utilizó el software RENO, éste requiere como entrada las dependencias funcionales y los atributos originales. Puesto que esta herramienta limita el número de dependencias a procesar, se decidió realizar la normalización por separado por cada uno de los subsistemas de la ACC. A modo de ejemplo, en la tabla 3 se muestran las relaciones obtenidas por RENO a partir de las dependencias funcionales de un subconjunto de los archivos del subsistema S-9 (Manejo de la información de los consumos de energía y su cobro).

Tabla 3. Algunas relaciones obtenidas con RENO para el subsistema S-9

R2: {MATERNO, NOMBRE1, NOMBRE2, PATERNO, RUT} CLAVE: [["RUT"]] {RUT------>NOMBRE2} {RUT------>NOMBRE1} {RUT------>MATERNO} {RUT------>PATERNO}
R3: {MOVCTA-CCOSTO, MOVCTA-CODCTA, MOVCTA-CTACORTA, MOVCTA-LOCAL, MOVCTA-NUMCLIE} CLAVE: [["MOVCTA-LOCAL","MOVCTA-CCOSTO","MOVCTA-CODCTA","MOVCTA-CTACORTA"]] {MOVCTA-CTACORTA, MOVCTA-CODCTA, MOVCTA-CCOSTO, MOVCTA-LOCAL----->MOVCTA-NUMCLIE}
R4: {MAE-COESTAD, MAE-MARCAME, MAE-MEDIDOR, MAE-NRUEDAS, MAE-NUFASES, MAE-POTEINS, MAE-PROP-MEDIDOR} CLAVE: [["MAE-MEDIDOR"]] {MAE-MEDIDOR------>MAE-POTEINS} {MAE-MEDIDOR------>MAE-NUFASES} {MAE-MEDIDOR------>MAE-NRUEDAS} {MAE-MEDIDOR------>MAE-MARCAME} {MAE-MEDIDOR------>MAE-COESTAD} {MAE-MEDIDOR------>MAE-PROP-MEDIDOR}

A partir de las relaciones obtenidas y sus correspondientes dependencias funcionales, y como última actividad del M-FF2RDB, se ha generado un esquema gráfico del modelo de base de datos para el subsistema, la Fig. 3 muestra parte de él.

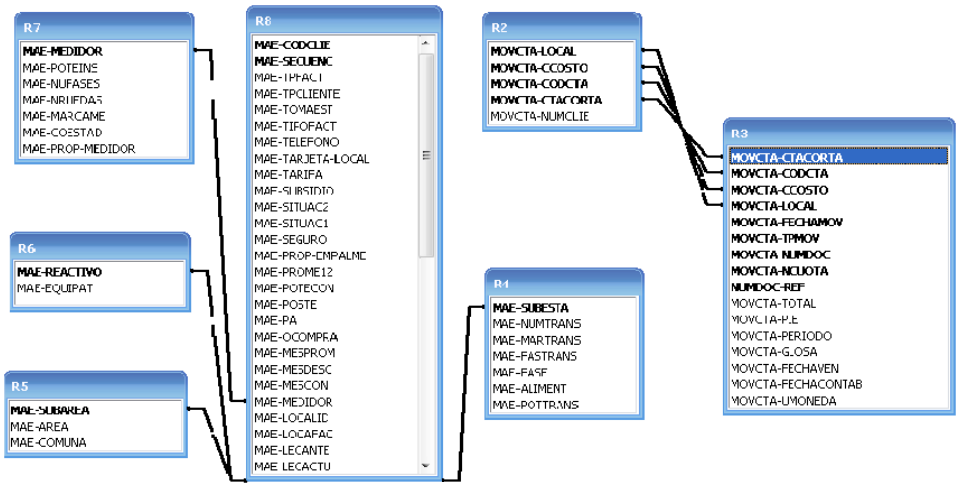

Fig. 3: Esquema parcial de la base de datos relacional del subsistema S-9.

4.4 Lecciones aprendidas

La aplicación del método M-FF2RDB ha favorecido el proceso transformación de un sistema de archivos heredado en un modelo de bases de datos relacional. A partir de la aplicación del caso de estudio se ha podido establecer que la puesta en operación del método supone la elección de las herramientas adecuadas al tamaño del problema. En este caso, la ACC resultó demasiado extensa. Estamos concientes que la última etapa es un proceso iterativo y probablemente deba ser aplicado como un subconjunto de tareas explicadas con mayor detalle. Los resultados de este caso de estudio, han motivado a Coopelan a continuar apoyando esta investigación, ciertos de que constituye un paso previo y esencial en la migración de sus datos heredados, primero, y sus LIS luego.

5 Conclusiones y Trabajos Futuros.

En este trabajo se ha presentado el primer paso conducente a resolver un problema real en empresas, que como Coopelan, poseen sistemas de información heredados extensos y esenciales para la operación del negocio. Hemos planteado una solución propia de la ingeniería que, de manera ordenada, nos permite enfrentar un problema relacionado con la creación de modelos de base de datos relacional a partir de archivos planos. Este paso es previo al proceso de migración de los datos y como tal constituye la base de un proceso de

mayor envergadura. Hemos mostrado una alternativa para solucionar este problema probando además nuestra propuesta en un caso real. El trabajo futuro está orientado a la evaluación de software ya construido o la construcción del mismo que simplifique la creación del modelo relacional, en cada una de las etapas del método. Por ejemplo, la etapa de "obtención de dependencias funcionales", puede ser complementada utilizando técnicas de minería de datos.

Agradecimientos. El caso de estudio se ha realizado gracias a la colaboración del señor Eduardo Robba encargado del área de computación de la Cooperativa Eléctrica Los Ángeles Limitada. El desarrollo de esta investigación es parte del proyecto N° 092119 2/R financiado por la Dirección de Investigación de la Universidad del Bio-Bio, Chile.

Referencias

1. Andersson, M. Extracting an Entity Relationship Schema from a Relational Database through Reverse Engineering. in 13th Int. Conference of the Entity Relationship Approach. 1994: Springer. p. 403-419.
2. Bisbal, J., Lawless, D., Wu, B., and Grimson, J., Legacy Information Systems: Issues and Directions. IEEE Software, 1999. 16(Septiembre/Octubre): p. 103-111.
3. Boronat, A., Pérez, J., Carsí, J., and Ramos, I., Two Experiences in Software Dynamics. Journal of Universal Computer Science, 2004. 10(4): p. 428-453.
4. Colosimo, M., De Lucia, A., Scanniello, G., and Tortora, G., Evaluating legacy system migration technologies through empirical studies. Information and Software Technology, 2009. Vol. 51, Issue 2: p. 433-447.
5. Chang-Yang, L., Migrating to Relational Systems: Problems, Methods, and Strategies. Contemporary Management Research, 2008. 4(4): p. 369-380.
6. De Lucia, A., Francese, R., Scanniello, G., and Tortora, G., Developing legacy system migration methods and tools for technology transfer. Software - Practice and Experience, 2008. Vol. 38 (13): p. 1333-1364.
7. De Miguel, A., Piattini, M., and Marcos, E., *Fundamentos y modelos de bases de datos*, ed. Editorial Ra-Ma. 1999.
8. Drumm, C., Schmitt, M., Do, H.-H., and Rahm, E. Quickmig: automatic schema matching for data migration projects. in Sixteenth ACM conference on Conference on information. 2007. Lisbon, Portugal: ACM. p. 107-116.
9. Elmasri, R. and Navathe, S., *Fundamentals of Database Systems*. 2003: Pearson/Addison Wesley.
10. Henrard, J., Cleve, A., and Hainaut, J.-L. Inverse Wrappers for Legacy Information Systems Migration. in First International Workshop on Wrapper Techniques for Legacy Systems (WRAP'04). 2004: TU Eindhoven Publish.,. p. 30–43.
11. Henrard, J., Roland, D., Cleve, A., and Hainaut, J.-L. An Industrial Experience Report on Legacy Data-Intensive System Migration. in International Conference on Software Maintenance, ICSM 2007. 2007. Paris, France: IEEE. p. 473-476.
12. Prather, J.C., Hales, J.W., Hage, M.L., Fehrs, S.J., and Hammond, W.E. Converting a legacy system database into relational format to enhance query efficiency. in Annual Symposium on Computer Applications in Medical Care. 1995. p. 372-376.